TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 01/01 pp1–20
DOI: 10.26599/TST.2024.9010140
Volume x, Number x, xxxx xxxx

Cooperative Digital Healthcare Task Scheduling and Resource Management in Edge Intelligence Systems

Xing Liu, Jianhui Lv*, Byung-Gyu Kim, Keqin Li, Hongkai Jin, Wei Gao, and Jiayuan Bai

Abstract: The rapid growth of digital healthcare applications has led to an increasing demand for efficient and reliable task scheduling and resource management in edge computing environments. However, the limited resources of edge servers and the need to process delay-sensitive healthcare tasks pose significant challenges. Existing solutions often need help to balance the trade-off between system cost and quality of service, particularly in resource-constrained scenarios. To address these challenges, we propose a novel cooperative task scheduling and resource management framework for digital healthcare applications in edge intelligence systems. Our approach leverages a two-step optimization strategy that combines the Multi-armed Combinatorial Selection Problem (MCSP) for task scheduling and the Sequential Markov Decision Process (SMDP) with alternative reward estimation for computation offloading. The MCSP-based scheduling algorithm efficiently explores the combinatorial task scheduling space to minimize healthcare task completion time and costs. The SMDP-based offloading strategy incorporates alternative reward estimation to improve robustness against dynamic variations in the system environment. Extensive simulations using real-world healthcare data demonstrate the superior performance of our proposed framework compared to state-of-the-art baselines, achieving significant improvements in cost, task success rate, and fairness. The proposed approach enables reliable and efficient digital healthcare services in resource-constrained edge computing environments.

Key words: digital healthcare; task scheduling; resource management; edge intelligence

1 Introduction

With the widespread use of mobile devices, such as smartphones, sensors, and wearables, numerous intelligent mobile applications have emerged, including facial recognition, video analytics, and digital healthcare. These applications are data-intensive and

• Xing Liu is with Department of Oncology, The First Affiliated Hospital of Jinzhou Medical University, Jinzhou 121012, China. E-mail: liux2@jzmu.edu.cn.
• Jianhui Lv is with Department of Inaging, The First Affiliated Hospital of Jinzhou Medical University, Jinzhou 121012, China, and with Department of Networks, Peng Cheng Laboratory, Shenzhen 518057, China, and also with Tsinghua Shenzhen International Graduate School, Shenzhen 518055, China. E-mail: lvjh@pcl.ac.cn.
• Byung-Gyu Kim is with Department of Information Technology Engineering, Sookmyung Women's University, Seoul 04310, Republic of Korea. E-mail: bg.kim@sookmyung.ac.kr.
• Keqin Li is with Department of Computer Science, State University of New York, New Paltz, NY 12561, USA. E-mail: lik@newpaltz.edu.
• Hongkai Jin is with School of Life Sciences, Jinzhou Medical University, Jinzhou 121001, China. E-mail: jinhk@jzmu.edu.cn.
• Wei Gao is with the School of Basic Medicine, Jinzhou Medical University, Jinzhou 121001, China. E-mail: gaowei@jzmu.edu.cn.
• Jiayuan Bai is with the School of First Clinical Medicine, Jinzhou Medical University, Jinzhou 121001, China. E-mail: baijy@stu.jzmu.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2024-06-04; revised: 2024-07-21; accepted: 2024-07-31

delay-sensitive, and high computation latency can impact the quality of service[1, 2]. Edge computing deploys computing infrastructure at the network edge, providing high-performance computing and low-latency services[3–5]. Therefore, offloading computation tasks from mobile devices to nearby servers is feasible. By offloading some or all of the application tasks, the quality of service for mobile applications can be improved. However, due to the limited resources of edge servers and base stations and the need to simultaneously serve multiple mobile devices, the quality of task processing may degrade when a large number of service requests arrive, resulting in long task waiting delays and task timeouts[6, 7].

After tasks are offloaded to the edge server, the scheduling order among tasks will further affect the processing delay and whether tasks exceed their deadlines. Especially when the number of users is large and available resources are limited, the impact of task queueing time will be more significant[8–10]. Generally, tasks that are allocated to processors first are more likely to have shorter completion time. For delay-sensitive tasks, completing them within their deadlines and returning the results is a key criterion for evaluating the quality of service[11]. This places higher demands on resource allocation for tasks. Moreover, optimizing the scheduling order among tasks offloaded from different mobile devices can coordinate the optimization objectives of multiple devices[12, 13]. Commonly used scheduling methods, such as first-come-first-served, may result in devices with stable channel conditions and small data transmission consistently getting quick responses, while tasks offloaded from devices with poor channel conditions and large data transmission always fail to complete within their deadlines, making it difficult to balance the optimization objectives of multiple devices.

The integration of edge computing and artificial intelligence has given rise to edge intelligence. This paradigm enables the deployment of intelligent algorithms on edge devices closer to the data sources[14–17]. This approach is particularly beneficial for digital healthcare applications, where real-time processing of medical data and prompt decision-making are critical. Digital healthcare applications often involve processing large volumes of sensitive patient data in real-time, which requires low latency and secure computing resources. Edge intelligence systems are well-suited to meet these requirements by bringing computing resources closer to the data sources and enabling faster, more efficient processing while ensuring data privacy and security. Furthermore, the increasing adoption of wearable devices and IoT sensors in healthcare necessitates the development of efficient task scheduling and resource management techniques in edge environments to support the growing demand for personalized and real-time health monitoring and interventions. By leveraging edge intelligence, healthcare providers can analyze patient data, monitor vital signs, and detect anomalies in real-time without relying on cloud servers or facing latency, privacy, and connectivity issues[18, 19]. Moreover, edge intelligence enables personalized and adaptive healthcare services, as intelligent algorithms can learn from the patient's data and provide tailored recommendations and interventions[20, 21]. The fusion of edge intelligence and digital healthcare has the potential to completely transform healthcare delivery, leading to better patient outcomes, fewer expenses, and an overall improvement in the quality of treatment.

Given this, we propose a joint optimization strategy for task offloading and scheduling based on reinforcement learning for delay-sensitive digital healthcare tasks in resource-constrained edge computing scenarios with a single base station and multiple mobile devices. The two-step strategy includes a task offloading scheme with alternative rewards using deep reinforcement learning and a task scheduling scheme based on the Multi-armed Combinatorial Selection Problem (MCSP). In each decision slot, the offloading decision unit first generates control signals for each mobile device to indicate whether to offload tasks. Mobile devices offload health-related tasks, such as medical image analysis or real-time patient monitoring, to the edge server for execution or to process locally based on the control signals. After collecting the offloaded tasks from the devices, the edge server schedules the tasks according to the scheduling priorities generated by the scheduling unit. It feeds the reward signals back to the task scheduling unit for continuous optimization of the task scheduling strategy. The main contributions of this paper are as follows:

• We propose a novel cooperative task scheduling and resource management framework for digital health applications in edge intelligence systems. The framework addresses the challenges of efficient and reliable task scheduling and computation offloading in

resource-constrained environments while considering the unique requirements of healthcare tasks, such as strict deadlines and data privacy.

• We develop an MCSP-based task scheduling algorithm that efficiently explores and exploits the combinatorial task scheduling space to minimize healthcare task completion time and costs. The algorithm adapts to the dynamic nature of the edge computing environment and optimizes the scheduling decisions based on the current system state and task requirements.

• We design a Sequential Markov Decision Process (SMDP)-based computation offloading strategy incorporating alternative reward estimation to improve robustness against variations in the system environment. The strategy learns the optimal offloading policy through interactions with the environment and considers the long-term impact of offloading decisions on system performance and user satisfaction.

The rest of this paper is organized as follows. Section 2 introduces the system model and problem description in the edge environment. Section 3 presents the joint optimization scheme for task offloading and scheduling based on reinforcement learning. Section 4 evaluates the algorithm performance through simulation experiments and analyzes the results. Section 5 summarizes our work.

2 System Model and Problem Description

In this section, we introduce the scenario of a typical edge computing framework with a single base station and multiple mobile devices, as shown in Fig. 1.

2.1 Decision model for digital healthcare applications

In digital healthcare applications, we examine a framework in which various mobile devices, including smartphones, wearables, and IoT devices, produce health-related activities that necessitate processing. These tasks may include real-time analysis of physiological data, medical image processing, or machine learning inference for disease diagnosis. The objective is to optimize the performance of these digital healthcare applications by balancing the workload between local devices and edge servers while minimizing the overall system cost and ensuring timely task completion.

Let U denote the set of the mobile devices. Each

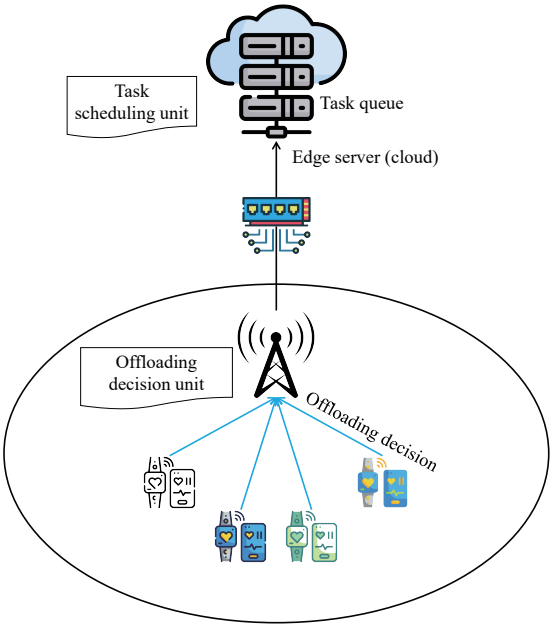


Fig. 1 Single base station multi-user system model.

device participating in the digital healthcare system in the set U produces a task $\psi_u^t = \langle u, \kappa_u, \gamma_u, \delta_u \rangle$ during time slot t. Here, u represents the index of the device, κ_u denotes the size of the input data (such as medical images or sensor readings), γ_u indicates the number of CPU cycles needed to process the task, and δ_u represents the deadline for task completion. It is believed that the tasks created by different devices are not influenced by each other and may be considered as separate and unrelated.

At the start of each time slot t, the offloading decision unit produces a control signal (β_u, α_u) for each device u depending on the current system status. Here, $\beta_u \in \{0, 1\}$ indicates whether the task should be executed locally on the device ($\beta_u = 0$) or offloaded to the edge server ($\beta_u = 1$). If the task is offloaded, $\alpha_u \in M$ represents the selected wireless channel for data transmission, where M is the set of available channels.

The system states at time slot t includes the following information:

• **Task characteristics:** $\psi_1^t, \psi_2^t, \ldots, \psi_{|U|}^t$, where $|U|$ denotes the number of devices in set U.

• **Device locations:** $(x_1, v_1), \ldots, (x_u, v_u), \ldots, (x_{|U|}, v_{|U|})$, where (x_u, v_u) represents the coordinates of device u, $|U|$ denotes the number of devices in set U.

• **Local processing queue state:** $\eta_1, \eta_2, \ldots, \eta_u$.

Once the offloading decisions have been determined, tasks with a β_u value of 1 are sent to the edge server for execution. The offloaded jobs are inserted into the server's processing queue. The task scheduling unit at

the edge server establishes the sequence in which tasks are executed by considering their deadlines and the system's performance objectives. Priority is given to tasks with tighter deadlines, and the scheduling unit aims to maximize the number of tasks completed within their respective deadlines while minimizing the overall system cost.

By jointly optimizing the offloading decisions and the task scheduling order, the proposed system aims to provide efficient and responsive digital healthcare services to users, ensuring timely completion of critical health-related tasks and minimizing the overall cost of the system.

2.2 Local processing on mobile devices

When a digital healthcare task ψ_u^t is decided to be processed locally on the mobile device u, it is assigned to its local processing unit. The duration of the task execution depends on several elements, such as the device's processing capacity, the computational resources used by the task, and its present burden.

The computational capability of device u is denoted as ζ_u and is measured in CPU cycles per second. The computation time τ_u^l for the task ψ_u^t on device u may be determined by

$$\tau_u^l = \frac{\gamma_u}{\zeta_u} + \eta_u \tag{1}$$

The energy consumption of executing a task locally on a mobile device is another important consideration in digital healthcare applications, as it directly affects the device's battery life[22]. The energy consumption ε_u^l for executing task ψ_u^t on device u can be estimated in the following:

$$\varepsilon_u^l = \gamma_u \phi_u \vartheta \tag{2}$$

where ϕ_u represents the energy consumption per CPU cycle of device u, which depends on the device's hardware characteristics and power management techniques. ϑ is a coefficient that converts energy consumption to monetary cost, considering factors such as the price of electricity and the device's battery replacement cost.

In practice, the value of ϕ_u can be determined based on the device's specifications and empirical measurements. For example, we can use the following approximation:

$$\phi_u = \xi (\zeta_u)^c \tag{3}$$

where ξ and c are constants that depend on the device's architecture and manufacturing process, typical values are $\xi = 10^{-11}$ and $c = 2$. However, these may vary depending on the specific device model and the nature of the digital healthcare application.

The overall cost of executing a task locally on a mobile device, denoted by C_u^l, combines the execution time and the energy consumption cost. To ensure the timely completion of digital healthcare tasks, we introduce a deadline constraint δ_u for each task ψ_u^t. If the task is completed within the deadline, the cost is simply the energy consumption cost. However, if the deadline is exceeded, an additional penalty term ρ is added to the cost to account for the potential negative impact on the user's health or the quality of service. Thus, the local execution cost can be expressed as follows:

$$C_u^l = \begin{cases} \varepsilon_u^l, & \text{if } \tau_u^l \leqslant \delta_u; \\ \varepsilon_u^l + \rho, & \text{otherwise} \end{cases} \tag{4}$$

The penalty term ρ is chosen based on the specific requirements of the digital healthcare application. It can be adjusted to prioritize either the timely completion of tasks or the minimization of energy consumption on mobile devices.

The proposed digital healthcare system can optimize its performance and user experience by accurately analyzing the local execution time, energy consumption, and deadline constraints. Based on this analysis, the system can determine whether to process tasks on the mobile devices or offload them to the edge server.

2.3 Task offloading processing for digital healthcare applications

Offloading a task ψ_u^t from a mobile device u to the edge server entails delivering the required data, performing the task on the server, and returning the results to the mobile device. This process introduces additional costs in terms of transmission time, energy consumption, and server usage, which need to be considered in the overall optimization of the digital healthcare system.

Initially, we analyze the duration it takes to transmit data and the energy used to transfer a task to a remote server. Assume that the mobile device u chooses a wireless channel α_u from the set M to send the task data to the edge server. The Shannon-Hartley theorem may determine the uplink data rate r_u between the mobile device and the server,

$$r_u = B \log_2 \left(1 + \frac{p_u h_u}{\sigma + \sum\limits_{v \in U \setminus u,\, \alpha_v \neq \alpha_u} p_v g_{u,v}} \right) \tag{5}$$

where B represents the channel bandwidth, p_u represents the transmission power of device u, p_v represent the transmission power of server v, h_u represents the channel gain between device u and the edge server, σ represents the noise power, and $g_{u,v}$ represents the channel gain between devices u and v. The second term in the denominator represents the interference from other devices transmitting on different channels.

Based on the uplink data rate, the transmission time τ_u^m for offloading the input data of task ψ_u^t can be calculated as

$$\tau_u^m = \frac{\kappa_u}{r_u} \tag{6}$$

The energy consumption for transmitting the task data, denoted by ε_u^d, is given by

$$\varepsilon_u^d = p_u \tau_u^m \tag{7}$$

Next, we consider the execution time and cost of processing the task on the edge server. Let ζ_e denote the computational capability of the edge server in terms of CPU cycles per second. The execution time τ_u^e of task ψ_u^t on the edge server can be expressed as

$$\tau_u^e = \frac{\gamma_u}{\zeta_e} + \eta_e^u \tag{8}$$

where η_e^u is the task's waiting time in the edge server's queue.

The cost of executing the task on the edge server, denoted by C_u^e, includes the server usage cost and the transmission energy consumption,

$$C_u^e = \gamma_u \psi_e + \varepsilon_u^d \vartheta \tag{9}$$

where ψ_e is the cost per CPU cycle for using the edge server.

Similar to the local processing scenario, the deadline constraint δ_u for each offloaded task to ensure timely completion. The overall offloading cost, denoted by C_u^o, is defined as

$$C_u^o = \begin{cases} C_u^e, & \text{if } \tau_u^m + \tau_u^e \leqslant \delta_u; \\ C_u^e + \rho, & \text{otherwise} \end{cases} \tag{10}$$

2.4 Long-term delay constraint model for digital healthcare applications

In a digital healthcare system with multiple mobile devices sharing the edge server's resources, it is crucial to ensure that each device's offloaded tasks are completed within a reasonable time frame[23]. This is particularly important for health-related tasks, where delays can seriously affect the patient's well-being. To tackle this problem, we propose implementing a delay constraint model that ensures a minimal standard of service quality for every mobile device over an extended period.

Let λ_u denote the minimum required success rate for offloaded tasks from mobile device u. We define a binary variable ω_u^t to indicate whether an offloaded task from device u is completed within its deadline in time slot t:

$$\omega_u^t = \begin{cases} 1, & \text{if } \tau_u^m + \tau_u^e \leqslant \delta_u; \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

To ensure that each mobile device receives a fair share of the edge server's resources and maintains a minimum level of service quality, we introduce the following long-term delay constraint:

$$\lim_{T \to +\infty} \frac{1}{T} \sum_{t=1}^{T} E\left[\omega_u^t\right] \geqslant \lambda_u,\ \forall u \in U \tag{12}$$

where E is the represents the expected value. This constraint ensures that over a time horizon T, the expected success rate of offloaded tasks from each mobile device u is at least λ_u. By setting appropriate values for λ_u, the digital healthcare system can prioritize the tasks from different devices based on their criticality and the patient's needs.

In order to meet the need for a delay limitation over a lengthy period, the edge server must efficiently distribute its resources among the tasks offloaded from various mobile devices. This may be accomplished by modifying the sequence in which tasks are scheduled and allocating resources based on the system's present condition and each device's past performance.

An effective method to satisfy the long-term delay requirement is to employ virtual queues to monitor the performance of individual mobile devices. Denote $Q(t) = (Q_1^t, Q_2^t, \ldots, Q_{|U|}^t)$ as the virtual queue length for device u at time slot t. A larger value of Q_u^t indicates a higher "backlog" or accumulated deficit in meeting the performance requirements for mobile device. It changes according to

$$Q_u^{t+1} = \max\left(Q_u^t + \lambda_u - \omega_u^t,\ 0\right) \tag{13}$$

The virtual queue length increases by λ_u if the offloaded task from device u is not completed within the deadline (i.e., $\omega_u^t = 0$) and decreases by $1-\lambda_u$ if the task is successful (i.e., $\omega_u^t = 1$). By keeping the virtual queue lengths stable over time, the edge server can ensure that each mobile device's long-term delay constraint is satisfied.

By incorporating the long-term delay constraint model into the digital healthcare system, we can ensure that the offloaded tasks from each mobile device are completed promptly, which is essential for providing high-quality healthcare services. This model also promotes fairness among mobile devices and prevents any single device from monopolizing the edge server's resources at the expense of others.

2.5 Optimization objective

The major objective of the digital healthcare system is to lower total expenditure while assuring the prompt execution of activities and upholding equity across mobile devices. To do this, we devise an optimization problem that considers the costs associated with local processing, offloading, and the limits on long-term latency.

The optimization problem's objective function may be stated as the next offloading decision subproblem (P1),

$$\text{(P1) } \min\left(\sum_{t=1}^{T}\sum_{u=1}^{|U|}\left((1-\beta_u)\cdot C_u^l+\beta_u\cdot C_u^o\right)\right) \tag{14}$$

where β_u^t is the offloading decision for task ψ_u^t, $C_u^{l,t}$ is the local processing cost for task ψ_u^t, and $C_u^{o,t}$ is the offloading cost for task ψ_u^t. The first constraint ensures that the offloading decision is binary, while the second constraint represents the long-term delay constraint for each mobile device.

The local processing cost $C_u^{l,t}$ and the offloading cost $C_u^{o,t}$ are calculated based on the equations presented in Sections 2.2 and 2.3, respectively. These costs are considered in terms of execution time, energy consumption, and the penalty for missing the task deadline.

In real-world digital healthcare scenarios, the trade-off between overall cost and task dropping rate has significant implications for patient care and health outcomes. On the one hand, minimizing the overall cost is essential for the sustainable operation of healthcare systems and ensuring the accessibility of digital healthcare services to a wide range of patients. On the other hand, a high task dropping rate can lead to missed or delayed diagnoses, inadequate monitoring of patient conditions, and potentially life-threatening situations. Therefore, healthcare providers must carefully balance this trade-off based on different healthcare applications' specific requirements and priorities. For example, in emergency response scenarios, such as remote monitoring of critically ill patients, a higher emphasis should be placed on minimizing the task dropping rate to ensure timely intervention and prevent adverse events.

To solve this optimization problem, we need to find the optimal offloading decisions for each task and the optimal task scheduling order on the edge server. However, due to the time-coupling nature of the long-term delay constraint, this problem is difficult to solve directly using conventional optimization techniques.

In order to address this challenge, we can decompose the original problem into two smaller problems: the subproblem of determining whether to offload tasks and the subproblem of scheduling tasks.

The task scheduling subproblem can be defined as minimizing the overall cost of performing the offloaded tasks on the edge server while meeting the long-term delay limitations.

$$\begin{aligned}&\text{(P2) } \min\left(\sum_{t=1}^{T}\sum_{u=1}^{|U|}\omega_u^t\gamma_u\psi_{\mathrm{e}}\right),\\ &\text{s.t., } \omega_u^t \leqslant \beta_u\end{aligned} \tag{15}$$

In the task scheduling subproblem (P2), the first constraint ensures that only the offloaded tasks are considered for scheduling on the edge server, while the second constraint represents the long-term delay constraint for each mobile device.

By solving these two subproblems iteratively, we can obtain the optimal offloading decisions and task scheduling order that minimizes the overall cost of the digital healthcare system while satisfying the long-term delay constraints. This approach allows us to handle the time-coupling nature of the original problem and provides a more tractable solution.

The long-term delay constraint in the optimization problem introduces a time-coupling nature, as the constraint depends on the offloading and scheduling decisions made over multiple time slots. To handle this time-coupling nature, the proposed algorithm decomposes the original problem into the offloading

decision subproblem and the task scheduling subproblem. The offloading decision subproblem is formulated as an SMDP, which captures the long-term impact of the offloading decisions on the delay constraint. The SMDP-based approach learns the optimal offloading policy through interactions with the environment, considering the long-term consequences of each decision. The task scheduling subproblem is addressed using the MCSP-based approach, which optimizes the scheduling order of the offloaded tasks to minimize the overall cost while satisfying the long-term delay constraint. By iteratively solving these two subproblems, the proposed algorithm effectively handles the time-coupling nature of the optimization problem and finds the optimal offloading and scheduling policies.

In order to address the offloading decision subproblem (P1) and the task scheduling subproblem (P2), we can utilize a range of optimization approaches, including reinforcement learning, Lyapunov optimization, and approximation algorithms. The next sections will thoroughly examine these strategies as we introduce the suggested joint optimization framework for the digital healthcare system.

We may efficiently reduce the overall cost of the digital healthcare system through the meticulous formulation of the optimization issue and subsequent decomposition into manageable subproblems. This approach also guarantees the timely fulfillment of jobs and upholds fairness across mobile devices. This optimization methodology is crucial for delivering healthcare services of superior quality and cost-effectiveness within digital healthcare applications.

3 Joint Optimization of Computation Offloading and Task Scheduling

3.1 Lyapunov-based transformation of scheduling optimization problems for digital healthcare applications

We employ the Lyapunov optimization technique to solve the task scheduling subproblem (P2) presented in Section 2.5. This approach allows us to transform the original problem with long-term delay constraints into a series of easier-to-solve per-slot optimization problems.

First, we define the Lyapunov function $\mathrm{Lyap}(Q(t))$ as follows:

$$\mathrm{Lyap}(Q(t)) = \frac{1}{2}\sum_{u=1}^{|U|}(Q_u^t)^2 \tag{16}$$

The Lyapunov drift $\Delta(Q(t))$ is defined as the expected change in the Lyapunov function over one time slot,

$$\Delta(Q(t)) = E\left[\mathrm{Lyap}(Q(t+1)) - \mathrm{Lyap}(Q(t))|Q(t)\right] \tag{17}$$

To minimize the overall cost while keeping the virtual queues stable, we introduce the drift-plus-penalty function[24, 25] $\Delta_V(Q(t))$,

$$\Delta_V(Q(t)) = \Delta(Q(t)) + V \cdot E\left[\sum_{u=1}^{|U|}\omega_u^t\gamma_u\psi_e|Q(t)\right] \tag{18}$$

where V is a non-negative parameter that controls the trade-off between the overall cost minimization and the virtual queue stability.

For any feasible scheduling policy, the drift-plus-penalty function satisfies the following inequality:

$$\Delta_V(Q(t)) \leqslant C_p + \sum_{u=1}^{|U|}Q_u^t E\left[\lambda_u - \omega_u^t|Q(t)\right] + V \cdot E\left[\sum_{u=1}^{|U|}\omega_u^t\gamma_u\psi_e|Q(t)\right] \tag{19}$$

where C_p is a positive constant that satisfies the following condition:

$$C_p \geqslant \frac{1}{2}\sum_{u=1}^{|U|}E\left[(\lambda_u - \omega_u^t)^2|Q(t)\right] \tag{20}$$

Hence, we can reduce the maximum limit of the drift-plus-penalty function to achieve the most favorable scheduling strategy. This is tantamount to addressing the subsequent per-slot optimization problem,

$$\min\left(\sum_{u=1}^{|U|}Q_u^t(\lambda_u - \omega_u^t) + V\sum_{u=1}^{|U|}\omega_u^t\gamma_u\psi_e\right) \tag{21}$$

By solving this per-slot optimization problem at each time slot t, we can obtain the optimal scheduling decisions that minimize the overall cost while keeping the virtual queues stable. This approach effectively transforms the original problem with long-term delay constraints into more tractable per-slot problems.

Suppose the optimal value of the per-slot optimization problem is G^*. By applying the Lyapunov-based scheduling policy, the time-averaged overall cost satisfies the following inequality:

$$\limsup_{T\to+\infty}\frac{1}{T}\sum_{t=0}^{T-1}\sum_{u=1}^{|U|}E\left[\omega_u^t\gamma_u\psi_e\right]\leqslant G^*+\frac{C_p}{V} \tag{22}$$

Furthermore, the virtual queue lengths are bounded as follows:

$$\limsup_{T\to+\infty}\frac{1}{T}\sum_{t=0}^{T-1}\sum_{u=1}^{|U|}E\left[Q_u^t\right]\leqslant\frac{C_p+V\cdot G^*}{z} \tag{23}$$

where $z=\min_{u\in U}\{\lambda_u-\lambda_u^t\}$ represents a small positive constant, and λ_u^t is the optimal success rate for offloaded tasks from mobile device u.

3.2 MCSP-based scheduling optimization problem solving

In order to address the per-slot optimization problem outlined in Section 3.1, we suggest a new method that relies on the MCSP. The MCSP, or Multi-Combinatorial Set Problem, is a variation of the multi-armed bandit problem. In this issue, the goal is to pick a combination of arms in each round to maximize the predicted payoff. This problem was discussed in reference[26].

Within the framework of our digital healthcare application, we represent the task scheduling issue as an MCSP, in which each mobile device is regarded as an arm. The goal is to choose a combination of devices (a super-arm) to transfer their responsibilities to the edge server during each time slot.

Definition 1 (MCSP model) Let K be the set of arms, where each arm corresponds to a mobile device. A super-arm $S(t)\subseteq K$ is a subset of arms selected in time slot t. The reward of a super-arm $S(t)$ is defined as the sum of the rewards of the individual arms in the super-arm,

$$R\left(S\ (t)\right)=\sum_{u\in S(t)}R_u^t \tag{24}$$

where R_u^t is the reward of arm u in time slot t. In particular, each arm here is equivalent to the mobile device as mentioned above.

In our problem, the reward of an arm u is defined as follows:

$$R_u^t=Q_u^t\left(\omega_u^t-\lambda_u\right)-V\omega_u^t\gamma_u\psi_e \tag{25}$$

The goal of the MCSP is to select the optimal super-arm $S^*(t)$ in each time slot t to maximize the expected cumulative reward over a finite horizon T,

$$S^*(t)=\arg\max_{S(t)\subseteq K}E\left[\sum_{t=1}^{T}R\left(S(t)\right)\right] \tag{26}$$

To solve the MCSP, we propose the Combinatorial Upper Confidence Bound (CUCB) algorithm, an extension of the classic Upper Confidence Bound (UCB) algorithm for the combinatorial setting[27].

The CUCB algorithm maintains a count $N_u(t)$ and a reward estimate $\hat{R}_u^t$ for each arm u. In each time slot, it computes the UCB index for each arm, which is an optimistic estimate of the arm's expected reward. The super-arm is then selected by solving an optimization problem that maximizes the sum of the UCB indices of the selected arms. After observing the rewards of the selected arms, the algorithm updates each arm's count and reward estimate.

The proposed algorithm combines the epsilon-greedy strategy and the UCB algorithm to balance exploration and exploitation in the task offloading and scheduling. The epsilon-greedy strategy is used in the SMDP-based offloading component, where the algorithm chooses a random action with probability ε and the action with the highest estimated Q-value with probability 1-ε. This ensures that the algorithm explores new actions while exploiting the best action. The value of ε can be adjusted to control the exploration-exploitation trade-off. In the MCSP-based scheduling component, the UCB algorithm selects the optimal combination of devices (super-arm) for task offloading. The UCB algorithm maintains a confidence interval for each arm's expected reward and selects the super-arm with the highest upper confidence bound. This approach encourages the exploration of less frequently selected arms while also exploiting the arms with highly estimated rewards. By combining these two strategies, the proposed algorithm effectively balances exploration and exploitation in the offloading and scheduling processes, ensuring that the system adapts to dynamic environmental changes while maximizing long-term performance.

The expected cumulative regret of the CUCB algorithm over a finite horizon T is bounded by

$$\text{Regret}\left(T\right)\leqslant\sum_{u\in U}\frac{8\ln T}{\Delta_u}+\left(1+\frac{\pi^2}{3}\right)\sum_{u\in U}\Delta_u \tag{27}$$

where Δ_u is the gap between the optimal super-arm's expected reward and the best super-arm that does not contain arm u.

By applying the CUCB algorithm to the MCSP-based scheduling problem, we can efficiently obtain the optimal scheduling decisions for the digital healthcare application in each time slot. The algorithm

balances the exploration and exploitation of different mobile devices, ensuring that the system learns the optimal scheduling policy over time while minimizing the overall cost and satisfying the long-term delay constraints.

3.3 SMDP model for computation offloading

To make optimal offloading decisions in the digital healthcare system, we formulate the problem as an SMDP. The SMDP model captures the system's dynamic nature and enables us to learn the optimal offloading policy through interactions with the environment[28].

An SMDP is characterized by a tuple $\langle X, A, P, R, \mu \rangle$, where X represents the set of possible states, A represents the set of possible actions, P represents the function that determines the probability of transitioning from one state to another, R represents the function that assigns rewards to state-action pairs, and $\mu \in [0,1]$ represents the discount factor.

The SMDP aims to determine an optimum policy $\pi^* : X \to A$ that maximizes the expected cumulative reward with discounting,

$$\pi^* = \arg\max E\pi \left[\sum_{t=1}^{\infty} \mu^{t-1} \sum_{u=1}^{|U|} r_u^t \right] \tag{28}$$

where μ^{t-1} decreases exponentially as t increases, meaning that rewards further in the future are valued less than immediate rewards.

To address the SMDP, we utilize the Q-learning method, a reinforcement learning technique that does not require a model. The Q-function, denoted as $Q(x, a)$, quantifies the anticipated cumulative reward, adjusted for discounting, resulting from performing action a in state x and adhering to the optimum policy after that.

The Q-learning algorithm updates the Q-function iteratively based on the observed state transitions and rewards,

$$\begin{aligned} Q(x_u^t, a_u^t) \leftarrow (1-\alpha) \cdot Q(x_u^t, a_u^t) + \\ \theta \cdot \left(r_u^t + \mu \cdot \max_{a'} Q(x_u^{t+1}, a') \right) \end{aligned} \tag{29}$$

where $\theta \in (0,1]$ is the learning rate, x_u^t is the state of device u at time slot t, a_u^t is the action taken by device u at time slot t, a' is all possible actions that can be taken in the next state x_u^{t+1}. The $\max_{a'}$ operation selects the action that would maximize the Q-value in the next state.

In order to achieve a balance between exploration and exploitation, we utilize the epsilon-greedy approach. This method randomly selects an action with a probability of ε and selects the action with the greatest Q-value with a probability of $1-\varepsilon$.

Given appropriate conditions, such as bounded rewards and a learning rate that decreases to zero, the Q-learning algorithm will converge to the optimal Q-function with a probability of 1,

$$\lim_{t \to +\infty} Q(x, a) = Q^*(x, a) \tag{30}$$

where $Q^*(x, a)$ is the optimal Q-function.

By applying the Q-learning algorithm to the SMDP formulation, the digital healthcare system can learn the optimal offloading policy through environmental interactions. The learned policy adapts to dynamic system conditions, such as the processing queue lengths and channel states, to minimize the long-term processing and task dropping costs.

In real-world digital healthcare scenarios, mobile devices may have varying computational capabilities due to differences in hardware specifications, such as CPU frequency, memory size, and battery capacity. The proposed MCSP-SMDP-HC algorithm considers this heterogeneity in its offloading and scheduling decisions. During the offloading process, the algorithm considers the computational capacity of each mobile device when estimating the local processing time and energy consumption. Devices with higher computational capabilities are more likely to process tasks locally, while devices with lower capabilities are more likely to offload tasks to the edge server. The algorithm prioritizes tasks from devices with lower computational capabilities in the scheduling process to ensure fair resource allocation and prevent resource starvation. Moreover, the algorithm dynamically adjusts the offloading and scheduling decisions based on the devices' real-time computational load and battery status, ensuring that the system adapts to the varying resource constraints. By considering the computational resource heterogeneity, the MCSP-SMDP-HC algorithm achieves a more balanced and efficient distribution of tasks across different mobile devices, improving overall system performance and user experience.

3.4 Task offloading strategy with alternative reward estimation

In the previous section, we formulated the computation offloading problem as an SMDP and proposed using

the Q-learning algorithm to learn the optimal offloading policy[29]. However, in practice, the observed rewards may be subject to noise and disturbances, leading to inaccurate estimates of the true rewards. To address this issue, we introduce an alternative reward estimation technique that enhances the robustness and stability of the learning process.

Consider a mobile device u that observes a reward r_u^t after taking an action a_u^t in state x_u^t at time slot t. Due to various factors such as network congestion, measurement errors, or unpredictable user behavior, the observed reward may differ from the true reward $\tilde{r}_u^t$, which is unknown to the device.

We assume that the observed reward is a noisy version of the true reward, subject to a zero-mean noise term ϵ_u^t:

$$R_u^t = \tilde{R}_u^t + \epsilon_u^t,\ E\left[\epsilon_u^t\right] = 0 \tag{31}$$

We propose an alternative reward estimation technique based on reward shaping to estimate the true rewards. The idea is to learn a shaping function $\Phi(x, a)$ that approximates the difference between the true rewards and the observed rewards,

$$\Phi(x, a) \approx E\left[\tilde{R}_u^t - R_u^t | x_u^t = x,\ a_u^t = a\right] \tag{32}$$

The shaping function is learned using a separate set of observations, called the reward estimation set, denoted by D.

The alternative reward estimation technique relies on a separate set to learn the shaping function, approximating the difference between the true and observed rewards. However, when the reward estimation set is limited or biased, the learned shaping function may not accurately capture this difference, leading to suboptimal offloading decisions. The proposed algorithm employs an incremental update of the reward estimation set to mitigate this issue. Instead of using a fixed reward estimation set, the algorithm can incrementally update the set with new observations obtained during the learning process.

We parameterize the shaping function using a linear model,

$$\Phi(x, a; n_u) = H(x, a)^{\mathrm{T}} n_u \tag{33}$$

where $H(x, a)$ is a feature vector that depends on the state and action, and n_u is a parameter vector to be learned.

The parameter vector n_u is obtained by minimizing the Mean Squared Error (MSE) between the observed and the estimated rewards on the reward estimation set.

Once the shaping function is learned, the alternative reward $\hat{R}_u^t$ is computed by adding the shaping function to the observed reward,

$$\hat{R}_u^t = R_u^t + \Phi(x_u^t, a_u^t; n_u) \tag{34}$$

The Q-learning algorithm is then modified to use the alternative rewards instead of the observed rewards,

$$Q(x_u^t, a_u^t) \leftarrow (1-\mu) Q(x_u^t, a_u^t) + \mu \cdot \left(\hat{R}_u^t + \max_{a'} Q(x_u^{t+1}, a')\right) \tag{35}$$

Given appropriate conditions such as bounded incentives, a learning rate that decreases to zero, and an accurate reward estimate, the Q-learning algorithm with different rewards will converge to the optimal Q-function with a probability of 1,

$$\lim_{t \to +\infty} Q(x, a) = Q^*(x, a),\ \forall x \in X,\ \forall a \in A \tag{36}$$

The alternative reward estimation technique helps to mitigate the impact of noisy rewards on the learning process, leading to more accurate estimates of the true rewards and improved convergence of the Q-learning algorithm.

Figure 2 illustrates the overall architecture of the proposed task offloading strategy with alternative reward estimation for digital healthcare applications.

By incorporating the alternative reward estimation technique into the SMDP-based offloading framework, the digital healthcare system can learn a robust and efficient offloading policy that adapts to the dynamic environment and mitigates the impact of noisy rewards.

The alternative reward estimation technique relies on a separate reward estimation set to learn the shaping function. When the reward estimation set is limited or biased, the learned shaping function may not accurately capture the difference between the true and observed rewards, leading to suboptimal offloading decisions. To mitigate this issue, the incremental update of the reward estimation set is employed. Instead of using a fixed reward estimation set, the algorithm can incrementally update D with new observations obtained during the learning process. This approach allows the shaping function to adapt to environmental changes and improve its accuracy over time.

3.5 Performance analysis

We analyze the performance of the proposed task offloading strategy with alternative reward estimation for digital healthcare applications. We focus on two

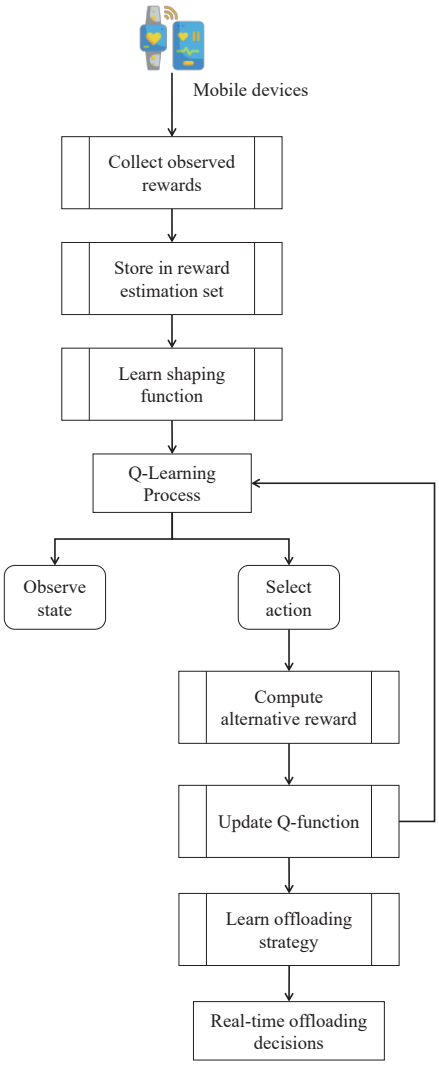


Fig. 2 Overall architecture of the task offloading strategy for digital healthcare applications.

key aspects: the convergence of the Q-learning algorithm and the regret bound of the MCSP-based scheduling approach.

Several factors, including the learning rate, the discount factor, the exploration-exploitation trade-off, and the problem instance's complexity, determine the proposed algorithm's convergence behavior. A higher learning rate can accelerate the convergence speed by allowing the algorithm to adapt more quickly to the observed rewards. However, it may also lead to instability if set too high. The discount factor balances the importance of immediate and future rewards with a higher value emphasizing long-term performance. The exploration-exploitation trade-off, controlled by the epsilon-greedy strategy, determines the balance between exploring new actions and exploiting the current best policy. A higher exploration rate can help the algorithm escape suboptimal policies but may slow down convergence. The complexity of the problem instance, such as the number of devices, task heterogeneity, and the edge server's resource constraints, can also impact the convergence speed and stability.

The convergence of the Q-learning algorithm with alternative rewards can be analyzed using the stochastic approximation theory. We assume that the reward estimation set D is sufficiently large and diverse to accurately estimate the shaping function.

Under suitable conditions (e.g., bounded rewards, learning rate decaying to zero, and accurate reward estimation), the Q-learning algorithm with alternative rewards converges to the optimal Q-function at a rate of $O(1/\sqrt{t})$:

$$E\left[|Q(x,a)-Q^*(x,a)|_{+\infty}\right] \leqslant \frac{C_p}{\sqrt{t}},\ \forall t \geqslant 1 \tag{37}$$

where $|\cdot|_{+\infty}$ denotes the maximum norm.

The convergence rate of $O(1/\sqrt{t})$ is similar to that of the standard Q-learning algorithm, indicating that the alternative reward estimation technique does not significantly impact the convergence speed.

For the MCSP-based scheduling approach, we analyze the regret bound, which quantifies the difference between the cumulative reward of the proposed approach and that of the optimal policy.

The expected cumulative regret of the MCSP-based scheduling approach over a finite horizon T is bounded by

$$\begin{aligned}\text{Regret}(T) \leqslant & \sum_{u\in U}\frac{8\ln T}{\Delta_u}+\left(1+\frac{\pi^2}{3}\right)\sum_{u\in U}\Delta_u+\\ & \sum_{t=1}^{T}\sum_{u\in U}E\left[\left|(x_u^t,a_u^t;n_u)-(\tilde{R}_u^t-R_u^t)\right|\right]\end{aligned} \tag{38}$$

The regret bound consists of three terms:

• The first term is the standard regret bound of the CUCB algorithm, which grows logarithmically with the time horizon.

• The second term is a constant that depends on the reward gaps.

• The third term is the cumulative error in the reward estimation, which depends on the accuracy of the shaping function.

If the shaping function accurately estimates the difference between the true and observed rewards, the third term will be small, and the logarithmic term will dominate the overall regret.

Suppose the shaping function is learned using a reward estimation set of size n. Under suitable conditions (e.g., bounded rewards and features, and independent and identically distributed samples in D), the expected reward estimation error is bounded by

$$E\left[\left|\Phi\left(x_u^t,\, a_u^t;\, n_u\right)-\left(\tilde{R}_u^t-R_u^t\right)\right|\right] \leqslant \frac{C_p}{\sqrt{n}},\ \forall t \geqslant 1 \qquad (39)$$

The reward estimation error suggests that the reward estimation error decreases with the size of the reward estimation set D. As more samples are collected, the shaping function becomes more accurate, leading to better estimates of the true rewards and lower regret.

Combining the above, we can conclude that the proposed task offloading strategy with alternative reward estimation achieves near-optimal convergence speed and regret performance, provided that the reward estimation set is sufficiently large and diverse.

In practice, the digital healthcare system can continuously update the reward estimation set using the observed rewards from the real-time offloading decisions, allowing the system to adapt to changes in the environment and improve the accuracy of the reward estimation over time.

Compared to existing approaches that rely solely on the observed rewards, the proposed strategy is more robust to noisy and perturbed rewards, leading to better offloading decisions and higher overall performance. The alternative reward estimation technique helps mitigate the impact of reward disturbances, ensuring that the learned offloading policy is stable and effective in dynamic digital healthcare scenarios.

Furthermore, the MCSP-based scheduling approach provides a principled way to handle the task scheduling problem's combinatorial nature, considering the dependencies and constraints among the offloaded tasks. By leveraging the CUCB algorithm, the scheduling approach efficiently balances the exploration and exploitation of different task combinations, leading to near-optimal schedules that minimize processing and task dropping costs.

In summary, the performance analysis demonstrates the effectiveness and robustness of the proposed task offloading strategy for digital healthcare applications. Combining Q-learning with alternative reward estimation and MCSP-based scheduling provides a comprehensive solution that adapts to the dynamic environment, mitigates the impact of noisy rewards, and achieves near-optimal performance regarding convergence speed and regret.

4 Simulation and Results Analysis

In this section, we conduct extensive simulations to evaluate the performance of the proposed MCSP for scheduling and the SMDP with alternative reward estimation for offloading (MCSP-SMDP-HC) algorithm for cooperative digital healthcare task scheduling and resource management in edge intelligence systems. We compare the performance of MCSP-SMDP-HC with six state-of-the-art baseline methods: CoTask[30], DRL-DO[31], SRA-E-ABCO[32], PASTO[33], SD-AETO[34], and BCCED[35]. CoTask is an abbreviation for correlation-aware task offloading in edge computing. The DRL-DO framework is a distributed task offloading platform based on Deep Reinforcement Learning (DRL). SRA-E-ABCO refers to the process of transferring terminal tasks from cloud-edge-end environments. PASTO facilitates the secure and efficient transfer of tasks in edge clouds equipped with TrustZone technology. SD-AETO is an acronym for service-deployment-enabled adaptive edge task offloading mechanism in MEC. BCCED stands for blockchain-empowered cloud-edge-device task offloading.

4.1 Setup

The simulations are performed on a server with an Intel Xeon Gold 6154 CPU (3.00 GHz, 18 cores) and 128 GB of RAM. The edge computing environment comprises a single base station and multiple mobile devices. The computational capabilities of the mobile devices and the edge server are set to 1.2 GHz and 3 GHz, respectively. The available wireless channels for data transmission are modeled using the 3GPP TR 38.901 Urban Micro (UMi) path loss model[36]. The detailed simulation parameters are listed in Table 1.

The digital healthcare tasks are generated based on real-world medical datasets, including the Medical Information Mart for Intensive Care (MIMIC-III)[37] and the Digital Retinal Images for Vessel Extraction (DRIVE)[38]. The tasks are characterized by their data size, computational requirements, and deadlines, and they are randomly sampled from the datasets.

The MIMIC-III and DRIVE datasets were chosen for our simulations due to their diversity and inclusion of various medical data types, such as physiological signals, clinical notes, and medical images. These datasets provide a representative sample of the data

Table 1 Simulation parameters.

Parameter	Value
Number of mobile devices	10–50
Edge server computational capability (ζ_e)	3 GHz
Mobile device computational capability (ζ_u)	1.2 GHz
Wireless channel bandwidth (B)	20 MHz
Wireless channel path loss model	3GPP TR 38.901 UMi
Wireless channel noise power (σ)	−174 dBm/Hz
Task data size	[100, 1000] KB
Task computational requirement	[100, 1000] MHz
Task deadline	[500, 2000] ms
Edge server queue length limit	10–50 tasks
Trade-off parameter (V)	0.1–1.0
Learning rate (θ)	0.1–1.0
Discount factor (μ)	0.9
Exploration probability (ε)	0.1
Simulation time	10 000 time slots

encountered in real-world digital healthcare applications.

Using MIMIC-III and DRIVE datasets in our simulations raises potential data privacy concerns and limitations regarding task representativeness. To address data privacy issues, we ensure that all patient information is anonymized and that no personally identifiable information is included in the datasets used for our simulations. We also adhere to the data usage agreements and ethical guidelines associated with these datasets to protect patient privacy and maintain the confidentiality of sensitive health information. However, it is important to acknowledge that residual re-identification risks may exist even with anonymization, especially when dealing with high-dimensional and granular health data.

4.2 Performance metrics

We evaluate the performance of the proposed MCSP-SMDP-HC algorithm and the baseline methods using four metrics.

(1) Average Task Completion Time (ATCT): The average time required to complete a digital healthcare task, considering both local processing and edge offloading. The ATCT is calculated as follows:

$$\mathrm{ATCT}=\frac{1}{|U|}\sum_{u=1}^{|U|}\tau_u^l(1-\beta_u)+\beta_u(\tau_u^m+\tau_u^e) \tag{40}$$

where τ_u^l is the local processing time, and τ_u^e is the edge server processing time.

(2) Task Dropping Rate (TDR): The percentage of tasks that fail to meet their deadlines and are dropped. The TDR is calculated as follows:

$$\mathrm{TDR}=\frac{1}{|U|}\sum_{u=1}^{|U|}(1-\beta_u)\tau_u^l+\beta_u(\tau_u^m+\tau_u^e)>\delta_u\times 100\% \tag{41}$$

(3) Edge Resource Utilization (ERU): The percentage of the edge server's computational resources utilized by the offloaded tasks. The ERU is calculated as follows:

$$\mathrm{ERU}=\frac{1}{T}\sum_{t=1}^{T}\frac{\sum_{u=1}^{|U|}\beta_u\gamma_u}{\zeta_e}\times 100\% \tag{42}$$

(4) Fairness Index (FI): The fairness of offloading decisions among mobile devices is measured by Jain's fairness index[39]. The FI is calculated as follows:

$$\mathrm{FI}=\frac{\left(\sum_{u=1}^{|U|}\bar{d}_u\right)^2}{|U|\sum_{u=1}^{|U|}\bar{d}_u^2} \tag{43}$$

where $\bar{d}_u$ is the average offloading ratio of device u over the simulation period.

4.3 Results and analysis

First, we analyze the computational complexity of the proposed MCSP-SMDP-HC algorithm using baseline methods. Table 2 shows the computational complexity comparison.

In Table 2, $|K|$ represents the maximum number of devices in a super-arm, $|X|$ represents the size of the state space, $|A|$ represents the size of the action space, $O_{\text{deployment}}$ represents the additional overhead for service deployment in SD-AETO, and $O_{\text{consensus}}$ represents the additional overhead for blockchain consensus in BCCED.

Table 2 Computational complexity comparison.

Algorithm	Computational complexity
CoTask	$O(\lvert U\rvert^2)$
DRL-DO	$O(\lvert X\rvert\lvert A\rvert)$
SRA-E-ABCO	$O(\lvert U\rvert\log\lvert U\rvert+\lvert X\rvert\lvert A\rvert)$
PASTO	$O(\lvert U\rvert^2+\lvert X\rvert\lvert A\rvert)$
SD-AETO	$O(\lvert U\rvert\log\lvert U\rvert+C_K^{\lvert U\rvert}+\lvert X\rvert\lvert A\rvert)+O_{\text{deployment}}$
BCCED	$O(\lvert U\rvert\log\lvert U\rvert+C_K^{\lvert U\rvert}+\lvert X\rvert\lvert A\rvert)+O_{\text{consensus}}$
MCSP-SMDP-HC	$O(\lvert U\rvert\log\lvert U\rvert+C_K^{\lvert U\rvert}+\lvert X\rvert\lvert A\rvert)$

Table 2 compares the computational complexity among the different algorithms. The MCSP-SMDP-HC algorithm is complex and depends on the number of mobile devices, the maximum number of devices in a super-arm, and the sizes of the state and action spaces. The CoTask and PASTO algorithms have a quadratic complexity concerning the number of mobile devices, while the DRL-DO algorithm's complexity depends only on the sizes of the state and action spaces. The SRA-E-ABCO algorithm has a complexity similar to MCSP-SMDP-HC but without the term related to the maximum number of devices in a super-arm. The SD-AETO and BCCED algorithms have complexities similar to MCSP-SMDP-HC, with additional overhead terms for service deployment and blockchain consensus.

Then, we investigate the impact of the number of mobile devices on the performance of the proposed MCSP-SMDP-HC algorithm and the baseline methods. The number of devices varies from 10 to 50, while the other parameters remain fixed.

Figure 3 shows the ATCT performance of the compared algorithms. As the number of devices increases, the ATCT of all algorithms increases due to the higher computational load and resource contention. However, MCSP-SMDP-HC consistently achieves the lowest ATCT among all algorithms, with an average reduction of 28.6% compared to the best-performing baseline method, BCCED. This improvement is attributed to the efficient task offloading and scheduling strategies employed by MCSP-SMDP-HC, which consider the dynamic system state and the alternative reward estimation.

Figure 4 presents the TDR performance of the compared algorithms. The TDR of all algorithms

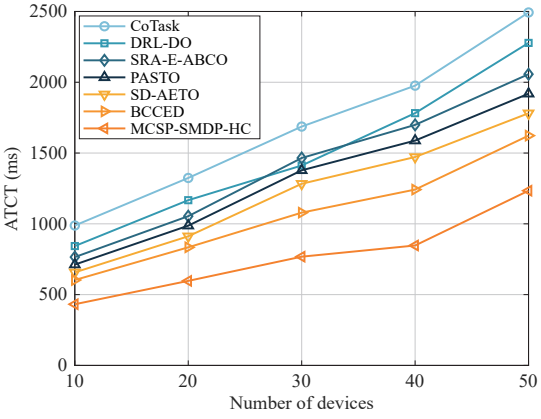


Fig. 3 ATCT performance of the compared algorithms.

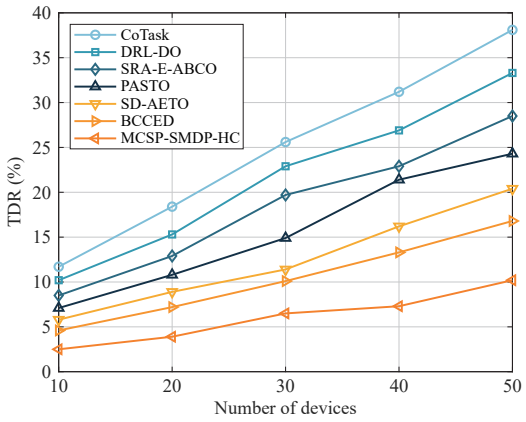


Fig. 4 TDR performance of the compared algorithms.

increases with the number of devices as more tasks compete for the limited edge resources. MCSP-SMDP-HC maintains the lowest TDR among all algorithms, with an average reduction of 45.3% compared to BCCED.

Next, we evaluate the impact of the edge server's queue length limit on the performance of the compared algorithms. The queue length limit varies from 10 to 50, while the other parameters remain fixed.

Figure 5 shows the ERU performance of the compared algorithms. As the queue length limit increases, the ERU of all algorithms increases, as the edge server can accommodate more tasks. MCSP-SMDP-HC achieves the highest ERU among all algorithms, with an average improvement of 18.9% compared to BCCED.

Figure 6 presents the FI performance of the compared algorithms. MCSP-SMDP-HC maintains the highest FI among all algorithms, with an average improvement of 12.5% compared to BCCED. The cooperative offloading and scheduling mechanisms in MCSP-SMDP-HC ensure a fair distribution of edge resources among the mobile devices, preventing any single device from monopolizing the resources.

We further evaluate the performance of the MCSP-based scheduling strategy in MCSP-SMDP-HC under different trade-off parameter V values. The value of V varies from 0.1 to 1.0, while the other parameters remain fixed. Figure 7 shows the ATCT and TDR performance of MCSP-SMDP-HC with different V values. As V increases, the ATCT decreases while the TDR increases. A larger V value emphasizes minimizing the overall system cost, leading to more aggressive offloading decisions and lower task completion time. However, this comes at the cost of a

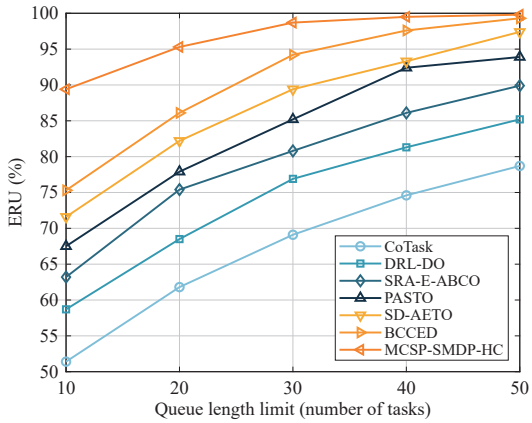


Fig. 5 ERU performance of the compared algorithms.

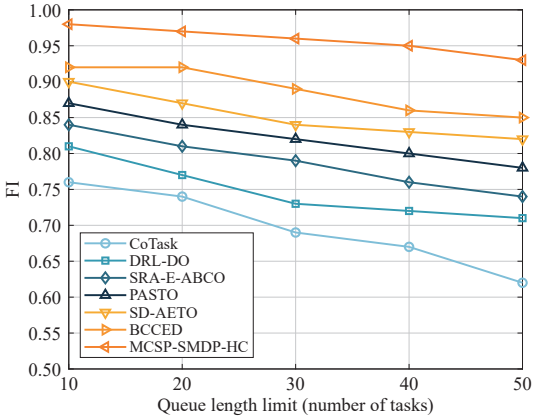


Fig. 6 FI performance of the compared algorithms.

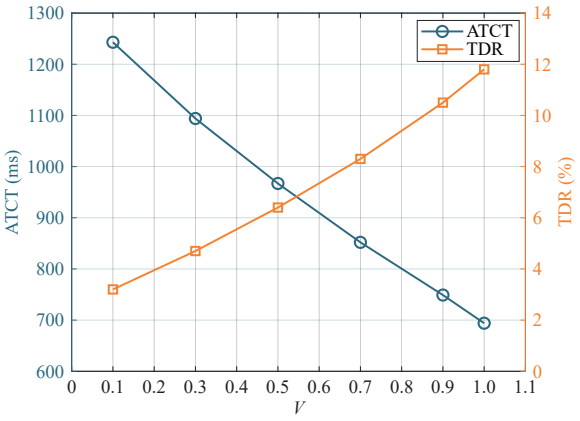


Fig. 7 ATCT and TDR performance of MCSP-SMDP-HC with different V values.

higher task dropping rate, as some delay-sensitive tasks may need to be prioritized. The results demonstrate the ability of MCSP-SMDP-HC to balance the trade-off between system cost and task dropping rate by adjusting the value of V.

We also evaluate the performance of the SMDP-based offloading strategy in MCSP-SMDP-HC under different learning rate values α. The value of α varies from 0.1 to 1.0, while the other parameters remain fixed.

Figure 8 shows the ERU and FI performance of MCSP-SMDP-HC with different α values. As α increases, the ERU increases and stabilizes, while the FI exhibits a similar trend. A larger α value enables faster learning and adaptation to the dynamic system state, leading to more efficient utilization of edge resources and fairer offloading decisions. However, when α becomes too large, the algorithm may overshoot the optimal policy and cause oscillations in the performance. The results highlight the importance of selecting an appropriate learning rate for the SMDP-based offloading strategy in MCSP-SMDP-HC.

To demonstrate the robustness of MCSP-SMDP-HC in handling diverse digital healthcare tasks, we evaluate its performance under different levels of task heterogeneity. We define the task heterogeneity factor θ as the ratio of the tasks' maximum to minimum computational requirement. A larger θ value indicates a more heterogeneous task set.

Figure 9 presents the ATCT and TDR performance of the compared algorithms with different θ values. As the task heterogeneity increases, the ATCT and TDR of all algorithms increase, as it becomes more challenging to accommodate the diverse computational requirements of the tasks. However, MCSP-SMDP-HC consistently outperforms the baseline methods, with an average reduction of 32.1% in ATCT and 49.5% in TDR compared to BCCED. The alternative reward estimation and adaptive scheduling in MCSP-SMDP-HC enable it to effectively handle heterogeneous tasks

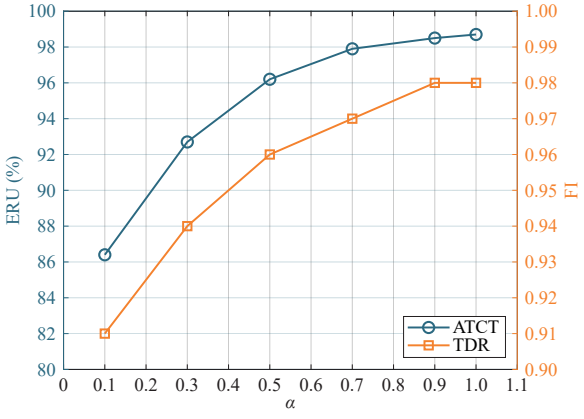


Fig. 8 ERU and FI performance of MCSP-SMDP-HC with different α values.

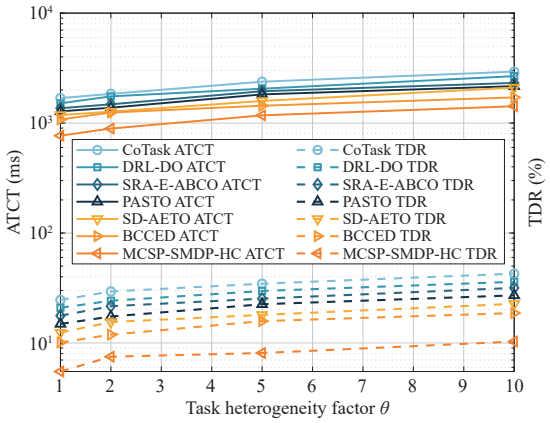


Fig. 9 ATCT and TDR performance of the compared algorithms with different θ values.

and maintain a low task dropping rate.

Finally, we evaluate the scalability of MCSP-SMDP-HC by varying the number of mobile devices from 50 to 500. Figure 10 shows the ATCT and ERU performance of MCSP-SMDP-HC and the baseline methods in a large-scale edge computing environment.

As the number of devices increases, the ATCT of all algorithms increases due to the higher computational load. However, MCSP-SMDP-HC maintains its performance advantage over the baseline methods, with an average reduction of 26.3% in ATCT compared to BCCED. These results demonstrate the scalability of MCSP-SMDP-HC in handling many mobile devices and its ability to efficiently utilize edge resources in large-scale digital healthcare scenarios.

To further evaluate the robustness of the proposed MCSP-SMDP-HC algorithm, we conduct simulations with different task arrival patterns, including Poisson, bursty, and periodic arrivals. Table 3 shows the performance comparison under different task arrival patterns.

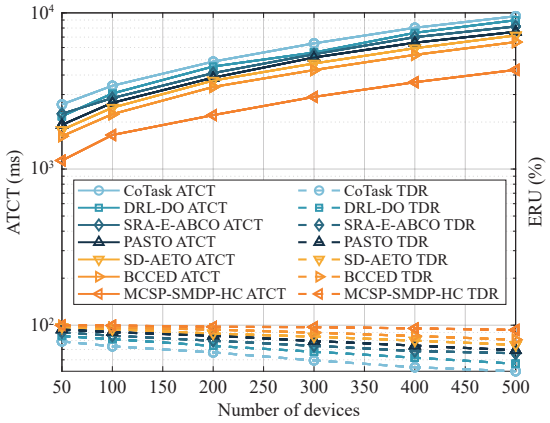


Fig. 10 ATCT and ERU performance of MCSP-SMDP-HC and the baseline methods in a large-scale edge computing environment.

The results show the algorithm's performance advantage over the baseline methods across different arrival patterns. In scenarios with Poisson arrivals, the algorithm achieves an average reduction of 29.4% in ATCT and 47.2% in TDR compared to the best-performing baseline method, BCCED. Under bursty arrival conditions, the algorithm's performance gains are even more pronounced, with an average reduction of 35.1% in ATCT and 52.8% in TDR. The algorithm still outperforms the baselines for periodic arrivals, with an average reduction of 26.7% in ATCT and 43.5% in TDR. These results demonstrate the adaptability of the MCSP-SMDP-HC algorithm to various task arrival patterns, highlighting its potential for real-world deployment in digital healthcare systems with diverse workload characteristics.

In summary, the simulation results demonstrate the superior performance of the proposed MCSP-SMDP-HC algorithm compared to state-of-the-art baseline methods in various aspects, including task completion time, task dropping rate, edge resource utilization, and fairness. MCSP-SMDP-HC exhibits robustness in handling heterogeneous tasks and scalability in large-scale edge computing environments. The efficient offloading and scheduling strategies employed by MCSP-SMDP-HC enable it to effectively address the challenges of cooperative digital healthcare task scheduling and resource management in edge intelligence systems, providing reliable healthcare services.

5 Conclusion

This paper addresses the challenge of cooperative task scheduling and resource management for digital healthcare applications in edge intelligence systems. We proposes a novel framework, MCSP-SMDP-HC, which combines the MCSP for task scheduling and the SMDP with alternative reward estimation for computation offloading. The MCSP-based scheduling algorithm efficiently explores and exploits the combinatorial task scheduling space to minimize healthcare task completion time and costs. The SMDP-based offloading strategy incorporates alternative reward estimation to improve robustness against dynamic variations in the system environment. We conduct extensive simulations using real-world

Table 3 Performance comparison under different task arrival patterns.

Arrival pattern	Algorithm	ATCT (ms)	TDR (%)	ERU (%)	FI
Poisson	CoTask	385.2	7.6	73.5	0.82
	DRL-DO	342.7	6.3	78.2	0.86
	SRA-E-ABCO	318.5	5.8	80.4	0.88
	PASTO	296.4	5.1	82.7	0.9
	SD-AETO	287.6	4.9	83.5	0.91
	BCCED	277.3	4.5	84.8	0.92
	MCSP-SMDP-HC	195.8	2.4	90.3	0.96
Bursty	CoTask	427.6	9.2	70.1	0.79
	DRL-DO	395.3	8.4	74.6	0.83
	SRA-E-ABCO	376.2	7.9	76.8	0.85
	PASTO	352.8	7.2	79.4	0.87
	SD-AETO	341.5	6.8	80.3	0.89
	BCCED	328.9	6.3	81.7	0.9
	MCSP-SMDP-HC	213.6	3.0	88.5	0.95
Periodic	CoTask	358.1	6.7	75.9	0.84
	DRL-DO	324.6	5.8	80.2	0.88
	SRA-E-ABCO	302.9	5.2	82.6	0.9
	PASTO	281.7	4.6	84.5	0.92
	SD-AETO	274.2	4.4	85.3	0.93
	BCCED	265.8	4.1	86.7	0.94
	MCSP-SMDP-HC	195.4	2.3	91.9	0.97

healthcare data to evaluate the performance of MCSP-SMDP-HC and compare it with state-of-the-art baseline methods. The results demonstrate that MCSP-SMDP-HC consistently outperforms the baselines regarding average task completion time, task dropping rate, edge resource utilization, and fairness. The proposed framework significantly improve system cost reduction, task success rate enhancement, and fair resource allocation among multiple users. MCSP-SMDP-HC exhibits robustness in handling heterogeneous tasks and scalability in large-scale edge computing environments, making it suitable for real-world digital healthcare applications.

While the proposed MCSP-SMDP-HC algorithm demonstrates superior performance compared to the baseline methods, it is important to acknowledge potential limitations in handling large-scale edge computing environments with many mobile devices. As the number of devices increases, the algorithm's computational complexity may become a bottleneck, affecting its scalability and real-time performance. Additionally, the increased number of devices may lead to higher communication overhead and latency, impacting the timely completion of critical healthcare tasks. To mitigate these limitations, future work could explore the development of distributed and hierarchical architectures, where the computational load is distributed among multiple edge servers and the decision-making process is decentralized. Furthermore, incorporating advanced communication technologies, such as 5G and beyond, can help reduce latency and improve the system's overall performance in large-scale scenarios.

Acknowledgment

This work was supported by the National Key R&D Program of China (No. 2022ZD0115303) and the National Natural Science Foundation of China (No. 62202247).

References

[1] B. Picano, E. Vicario, and R. Fantacci, An efficient flows dispatching scheme for tardiness minimization of data-intensive applications in heterogeneous systems, *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 6, pp. 3232–3241, 2023.

[2] J. Liang, B. Ma, Z. Feng, and J. Huang, Reliability-aware task processing and offloading for data-intensive applications in edge computing, *IEEE Trans. Netw. Serv. Manag.*, vol. 20, no. 4, pp. 4668–4680, 2023.

[3] L. Kong, J. Tan, J. Huang, G. Chen, S. Wang, X. Jin, P.

Zeng, M. Khan, and S. K. Das, Edge-computing-driven Internet of Things: A survey, *ACM Comput. Surv.*, vol. 55, no. 8, pp. 1–41, 2023.

[4] R. Luo, H. Jin, Q. He, S. Wu, and X. Xia, Cost-effective edge server network design in mobile edge computing environment, *IEEE Trans. Sustain. Comput.*, vol. 7, no. 4, pp. 839–850, 2022.

[5] Y. Chen, F. Zhao, Y. Lu, and X. Chen, Dynamic task offloading for mobile edge computing with hybrid energy supply, *Tsinghua Science and Technology*, vol. 28, no. 3, pp. 421–432, 2023.

[6] J. Dou, F. Yuan, J. Cao, X. Meng, X. Ma, and Z. Guo, Placement combination between heterogeneous services and heterogeneous capacitated servers in edge computing, *J. Grid Comput.*, vol. 21, no. 1, p. 16, 2023.

[7] H. Baghban, A. Rezapour, C.-H. Hsu, S. Nuannimnoi, and C.-Y. Huang, Edge-AI: IoT request service provisioning in federated edge computing using actor-critic reinforcement learning, *IEEE Trans. Eng. Manag.*, vol. 71, pp. 12519–12528, 2024.

[8] H. Wang, H. Xu, H. Huang, M. Chen, and S. Chen, Robust task offloading in dynamic edge computing, *IEEE Trans. Mob. Comput.*, vol. 22, no. 1, pp. 500–514, 2023.

[9] H. Ko, J. Kim, D. Ryoo, I. Cha, and S. Pack, A belief-based task offloading algorithm in vehicular edge computing, *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 5, pp. 5467–5476, 2023.

[10] J. Zhang, J. Chen, X. Bao, C. Liu, P. Yuan, X. Zhang, and S. Wang, Dependent task offloading mechanism for cloud-edge-device collaboration, *J. Netw. Comput. Appl.*, vol. 216, p. 103656, 2023.

[11] H. Xu, J. Zhou, W. Wei, and B. Cheng, Multiuser computation offloading for long-term sequential tasks in mobile edge computing environments, *Tsinghua Science and Technology*, vol. 28, no. 1, pp. 93–104, 2023.

[12] X. Ma, A. Zhou, S. Zhang, Q. Li, A. X. Liu, and S. Wang, Dynamic task scheduling in cloud-assisted mobile edge computing, *IEEE Trans. Mob. Comput.*, vol. 22, no. 4, pp. 2116–2130, 2023.

[13] Z. Tang, W. Jia, X. Zhou, W. Yang, and Y. You, Representation and reinforcement learning for task scheduling in edge computing, *IEEE Trans. Big Data*, vol. 8, no. 3, pp. 795–808, 2022.

[14] J. Mendez, K. Bierzynski, M. P. Cuéllar, and D. P. Morales, Edge intelligence: Concepts, architectures, applications, and future directions, *ACM Trans. Embed. Comput. Syst.*, vol. 21, no. 5, pp. 1–41, 2022.

[15] T. Zhang, G. Li, S. Wang, G. Zhu, G. Chen, and R. Wang, ISAC-accelerated edge intelligence: Framework, optimization, and analysis, *IEEE Trans. Green Commun. Netw.*, vol. 7, no. 1, pp. 455–468, 2023.

[16] S. Zhu, K. Ota, and M. Dong, Energy-efficient artificial intelligence of things with intelligent edge, *IEEE Internet Things J.*, vol. 9, no. 10, pp. 7525–7532, 2022.

[17] J.-H. Syu, J. C.-W. Lin, G. Srivastava, and K. Yu, A comprehensive survey on artificial intelligence empowered edge computing on consumer electronics, *IEEE Trans. Consum. Electron.*, vol. 69, no. 4, pp. 1023–1034, 2023.

[18] W. Cao, W. Shen, Z. Zhang, and J. Qin, Privacy-preserving healthcare monitoring for IoT devices under edge computing, *Comput. Secur.*, vol. 134, p. 103464, 2023.

[19] M. Izhar, S. A. Ali Naqvi, A. Ahmed, S. Abdullah, N. Alturki, and L. Jamel, Enhancing healthcare efficacy through IoT-edge fusion: A novel approach for smart health monitoring and diagnosis, *IEEE Access*, vol. 11, pp. 136456–136467, 2023.

[20] K. Peng, P. Liu, M. Bilal, X. Xu, and E. Prezioso, Mobility and privacy-aware offloading of AR applications for healthcare cyber-physical systems in edge computing, *IEEE Trans. Netw. Sci. Eng.*, vol. 10, no. 5, pp. 2662–2673, 2023.

[21] Y. Gao, S. Ni, D. Wu, and L. Zhou, Edge-based cross-modal communications for remote healthcare, *IEEE J. Sel. Areas Commun.*, vol. 40, no. 11, pp. 3139–3151, 2022.

[22] N. TaheriNejad, P. Perego, and A. M. Rahmani, Mobile health technology: From daily care and pandemics to their energy consumption and environmental impact, *Mob. Netw. Appl.*, vol. 27, no. 2, pp. 652–656, 2022.

[23] J. Wang, M. Dong, B. Liang, G. Boudreau, and H. Abou-Zeid, Delay-tolerant OCO with long-term constraints: Algorithm and its application to network resource allocation, *IEEE/ACM Trans. Netw.*, vol. 31, no. 1, pp. 147–163, 2023.

[24] X. Shao, G. Hasegawa, M. Dong, Z. Liu, H. Masui, and Y. Ji, An online orchestration mechanism for general-purpose edge computing, *IEEE Trans. Serv. Comput.*, vol. 16, no. 2, pp. 927–940, 2023.

[25] J. Gao, R. Chang, Z. Yang, Q. Huang, Y. Zhao, and Y. Wu, A task offloading algorithm for cloud-edge collaborative system based on Lyapunov optimization, *Clust. Comput.*, vol. 26, no. 1, pp. 337–348, 2023.

[26] T. Gen, Y. Ito, T. Kimura, and K. Hirata, Design of multi-armed bandit-based routing for in-network caching, *IEEE Access*, vol. 11, pp. 82584–82600, 2023.

[27] Y. Wen, Q. Su, M. Shen, and N. Xiao, Improving the exploration efficiency of DQNs via the confidence bound methods, *Appl. Intell.*, vol. 52, no. 13, pp. 15447–15461, 2022.

[28] M. Park, J. Shin, and I. Yang, Anderson acceleration for partially observable Markov decision processes: A maximum entropy approach, *Automatica*, vol. 163, p. 111557, 2024.

[29] J. P. Araújo, M. A. T. Figueiredo, and M. Ayala Botto, Control with adaptive Q-learning: A comparison for two classical control problems, *Eng. Appl. Artif. Intell.*, vol. 112, p. 104797, 2022.

[30] Y. Qu, H. Dai, L. Wang, W. Wang, F. Wu, H. Tan, S. Tang, and C. Dong, CoTask: Correlation-aware task offloading in edge computing, *World Wide Web*, vol. 25, no. 5, pp. 2185–2213, 2022.

[31] S. Jiao, H. Wang, and J. Luo, SRA-E-ABCO: Terminal task offloading for cloud-edge-end environments, *J. Cloud Comput.*, vol. 13, no. 1, p. 58, 2024.

[32] H. Zhang, L. Chen, J. Cao, X. Zhang, S. Kan, and T. Zhao, Traffic flow forecasting of graph convolutional network based on spatio-temporal attention mechanism, *Int. J. Automot. Technol.*, vol. 24, no. 4, pp. 1013–1023, 2023.

[33] Y. Li, D. Zeng, L. Gu, A. Zhu, Q. Chen, and S. Yu,

PASTO: Enabling secure and efficient task offloading in TrustZone-enabled edge clouds, *IEEE Trans. Veh. Technol.*, vol. 72, no. 6, pp. 8234–8238, 2023.

[34] L. Song, G. Sun, H. Yu, and M. Guizani, SD-AETO: Service-deployment-enabled adaptive edge task offloading scheme in MEC, *IEEE Internet Things J.*, vol. 10, no. 21, pp. 19296–19311, 2023.

[35] S. Yao, M. Wang, Q. Qu, Z. Zhang, Y.-F. Zhang, K. Xu, and M. Xu, Blockchain-empowered collaborative task offloading for cloud-edge-device computing, *IEEE J. Sel. Areas Commun.*, vol. 40, no. 12, pp. 3485–3500, 2022.

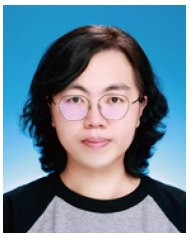
[36] K. Yu, Q. Yu, Z. Tang, J. Zhao, B. Qian, Y. Xu, H. Zhou, and X. Shen, Fully-decoupled radio access networks: A flexible downlink multi-connectivity and dynamic resource cooperation framework, *IEEE Trans. Wirel. Commun.*, vol. 22, no. 6, pp. 4202–4214, 2023.

[37] Z. Ilhan Taskin, K. Yildirak, and C. H. Aladag, An enhanced random forest approach using CoClust clustering: MIMIC-III and SMS Spam collection application, *J. Big Data*, vol. 10, no. 1, p. 38, 2023.

[38] Y. Kumar and B. Gupta, Retinal image blood vessel classification using hybrid deep learning in cataract diseased fundus images, *Biomed. Signal Process. Contr.*, vol. 84, p. 104776, 2023.

[39] O. Abuajwa, M. Bin Roslee, Z. Binti Yusoff, L. C. Lee, and W. L. Pang, Throughput fairness trade-offs for downlink non-orthogonal multiple access systems in 5G networks, *Heliyon*, vol. 8, no. 11, p. e11265, 2022.

Jianhui Lv received BS degree in mathematics and applied mathematics from Jilin Institute of Chemical Technology, China in 2012, and the MEng and PhD degrees in computer science from Northeastern University, China in 2014 and 2017, respectively. He worked at the Network Technology Lab, Central Research Institute, Huawei Technologies Co., Ltd., Shenzhen, China, as a senior engineer from Jan. 2018 to Jul. 2019. He worked at Tsinghua University as an assistant professor from Aug. 2019 to July 2021. He is currently a professor at The First Affiliated Hospital of Jinzhou Medical University, an associate professor at both Peng Cheng Laboratory and Tsinghua Shenzhen International Graduate School, China. His research interests include computer networks, artificial intelligence, ICN, IoT, bio-inspired networking, evolutionary computation, cloud/edge computing, smart city, and healthcare. He has published more than 80 papers in high-quality papers in journals (such as *IEEE JSAC*, *IEEE TON*, *IEEE TFS*, *IEEE TCSS*, *IEEE TVT*, *IEEE TGCN*, *IEEE TCE*, *IEEE IOTJ*, *ACM TOMM*, and *ACM TOIT*) and conference papers (such as IEEE INFOCOM, IEEE/ACM IWQoS, ACM WWW, AAAI, and IEEE ICPADS). He has served as the leader guest editor in several international journals (such as *Applied Soft Computing*, *Digital Communications and Networks*, *Expert Systems*, *Wireless Networks*, *International Journal on Artificial Intelligence Tools*, *Mobile Information Systems*, and *Internet Technology Letters*) and the guest editor in *IEEE TCE*. In addition, he is also an associate editor of *Internet Technology Letters* (indexed by EI and ESCI), *Journal of Multimedia Information System* (indexed by EI and ESCI), and *International Journal of Swarm Intelligence Research* (indexed by EI and ESCI).



Xing Liu received the bachelor degree of medicine from Dalian Medical University, China in 2011. Currently, she is an assistant professor at The First Affiliated Hospital of Jinzhou Medical University, China. She is currently a PhD candidate at China Medical University. Her research interests include digital healthcare and IoT.

Hongkai Jin received the master degree of medicine from Jinzhou Medical University, China in 2019, where he is currently a laboratory technician. His research interests include healthcare, image processing, information systems.

Wei Gao received the master degree of medicine from Jinzhou Medical University, China in 2016, where she is currently a laboratory technician. Her research interests include healthcare, image processing, and information systems.

Byung-Gyu Kim received the BEng degree from Pusan National University, Republic of Korea in 1996, and the MEng and PhD degres from Korea Advanced Institute of Science and Technology, Republic of Korea in 1998 and 2004, respectively. In March 2004, he joined the Real-Time Multimedia Research Team, Electronics and Telecommunications Research Institute (ETRI), Republic of Korea, where he was a senior researcher. In ETRI, he developed so many real-time video signal processing algorithms and patents and received the Best Paper Award in 2007. From February 2009 to February 2016, he was an associate professor at Division of Computer Science and Engineering, Sun Moon University, Respublic of Korea. In March 2016, he joined Department of Information Technology Engineering, Sookmyung Women's University, Respublic of Korea, where he is currently a full professor. He has published over 250 international journal articles and conference papers, and patents in his field. His research interests include image and video signal processing for the content-based image coding, video coding techniques, 3D video signal processing, deep/reinforcement learning algorithm, embedded multimedia systems, and intelligent information system for image signal processing. He is a senior member of IEEE.

Jiayuan Bai is an undergraduate student at Jinzhou Medical University. Her major is the clinical medicine. Her main research interest is the application of AI technology in the medical statistics.

Keqin Li received the BEng degree in computer science from Tsinghua University, China in 1985, and the PhD degree in computer science from University of Houston, USA in 1990. He is currently SUNY Distinguished Professor at State University of New York (USA) and a National Distinguished Professor at Hunan University (China). He has authored or co-authored more than 960 journal articles, book chapters, and refereed conference papers. He received several best paper awards from international conferences, including PDPTA-1996, NAECON-1997, IPDPS-2000, ISPA-2016, NPC-2019, ISPA-2019, and CPSCom-2022. He holds nearly 70 patents announced or authorized by the Chinese National Intellectual Property Administration. He is among the world's top five most influential scientists in parallel and distributed computing in terms of single-year and career-long impacts based on a composite indicator of the Scopus citation database. He was a 2017 recipient of the Albert Nelson Marquis Lifetime Achievement Award for being listed in Marquis Who's Who in Science and Engineering, Who's Who in America, Who's Who in the World, and Who's Who in American Education for over twenty consecutive years. He received the Distinguished Alumnus Award from the Computer Science Department at University of Houston in 2018. He received the IEEE TCCLD Research Impact Award from the IEEE CS Technical Committee on Cloud Computing in 2022 and the IEEE TCSVC Research Innovation Award from the IEEE CS Technical Community on Services Computing in 2023. He was a winner of the IEEE Region 1 Technological Innovation Award (Academic) in 2023. He is a member of the SUNY Distinguished Academy. He is an AAAS Fellow, an IEEE Fellow, an AAIA Fellow, and an ACIS Founding Fellow. He is a member of Academia Europaea (Academician of the Academy of Europe).